(12) UK Patent Application (19) GB (11) 2 266 032 (13) A

(43) Date of A publication 13.10.1993

(21) Application No 9205094.7

(22) Date of filing 09.03.1992

(51) INT CL[5]
H04L 12/40

(52) UK CL (Edition L)
H4P PPNC

(56) Documents cited
US 3978451 A

(58) Field of search
UK CL (Edition K) H4P PPNC
INT CL[5] G06F 13/14, H04L 12/40

(71) Applicant
Racal-Datacom Limited

(Incorporated in the United Kingdom)

Landata House, Station Road, Hook, Hampshire, RG27 9PE, United Kingdom

(72) Inventors
John Hill Boal
Mark Kendal Newton

(74) Agent and/or Address for Service
Reddie & Grose
16 Theobalds Road, London, WC1X 8PL, United Kingdom

(54) Computer communications bus and controller

(57) A packet handler (11) controls communications between a processor bus (10) and a packet bus comprising 16 information lines INFO-15, a single control line NOT-STROBE and a clock line CLOCK. The information lines can carry not only data DATAO-15 but also arbitration signals ARBO to ARB++, PRIO-7. To permit this multiplexed operation and tight control over the bus states, the states follow precisely defined sequences demarcated by state change codes. A state change is signalled by NOT-STROBE going low with simultaneous placement of a state change code DELIMO-3 on the lines INFO-3. Moreover the state change strobe is validated by a common qualifier QUALIFO-11 on information lines INFO-15. The packet handler (11) and bust (10) are particularly suited to communications using packets comprising typically header, header reply, data and data reply frames. The transmitter sends the header and awaits an acknowledgement header replay generated by the receiver packet handler with little or no host intervention. According to the reply (NAK, WAK, ACK) the transmitter can abort the packet, wait, or immediately send any data frame portion of the packet. A further reply may be sent confirming integrity of the received packet.

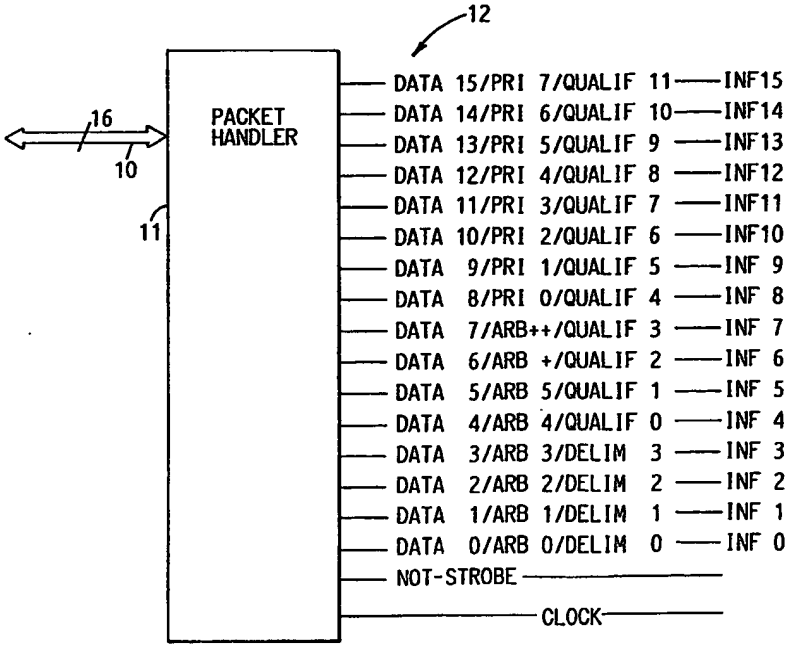


FIG.5

GB 2 266 032 A

At least one drawing originally filed was informal and the print reproduced here is taken from a later filed formal copy.

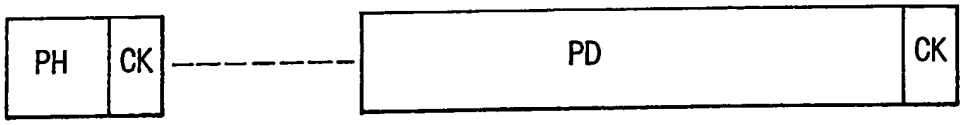


FIG. 1

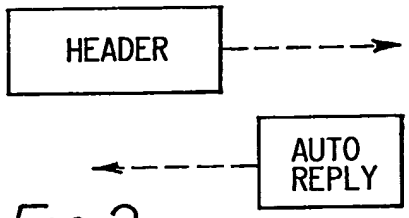


FIG. 2

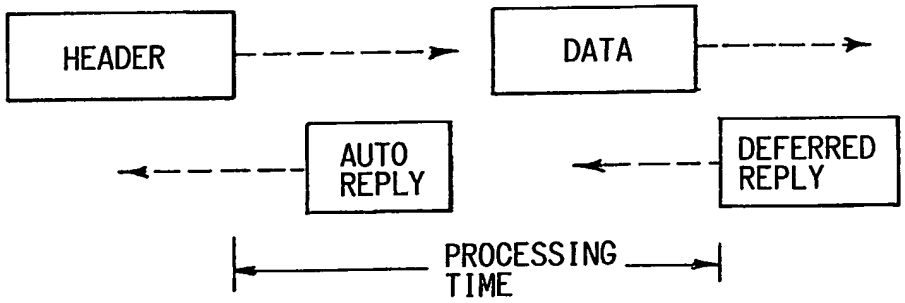


FIG. 3

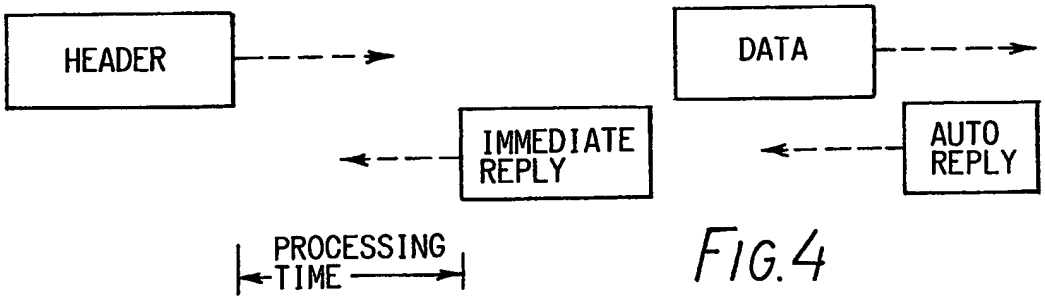


FIG. 4

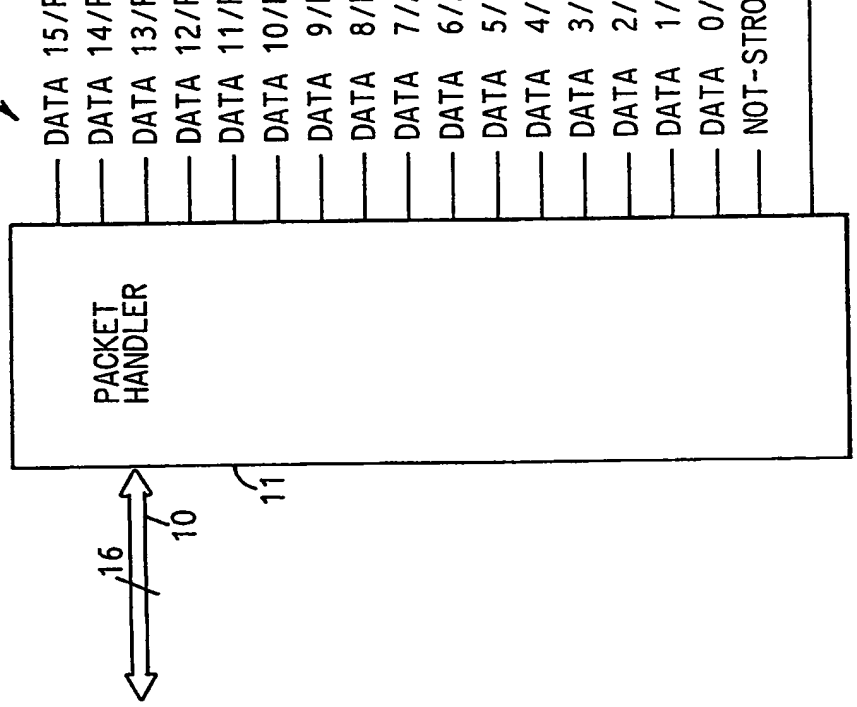

12
PACKET HANDLER
DATA 15/PRI 7/QUALIF 11
DATA 14/PRI 6/QUALIF 10
DATA 13/PRI 5/QUALIF 9
DATA 12/PRI 4/QUALIF 8
DATA 11/PRI 3/QUALIF 7
DATA 10/PRI 2/QUALIF 6
DATA 9/PRI 1/QUALIF 5
DATA 8/PRI 0/QUALIF 4
DATA 7/ARB++/QUALIF 3
DATA 6/ARB +/QUALIF 2
DATA 5/ARB 5/QUALIF 1
DATA 4/ARB 4/QUALIF 0
DATA 3/ARB 3/DELIM 3
DATA 2/ARB 2/DELIM 2
DATA 1/ARB 1/DELIM 1
DATA 0/ARB 0/DELIM 0
NOT-STROBE
CLOCK
INF15
INF14
INF13
INF12
INF11
INF10
INF 9
INF 8
INF 7
INF 6
INF 5
INF 4
INF 3
INF 2
INF 1
INF 0
11
10
16

FIG.5

FIG.6

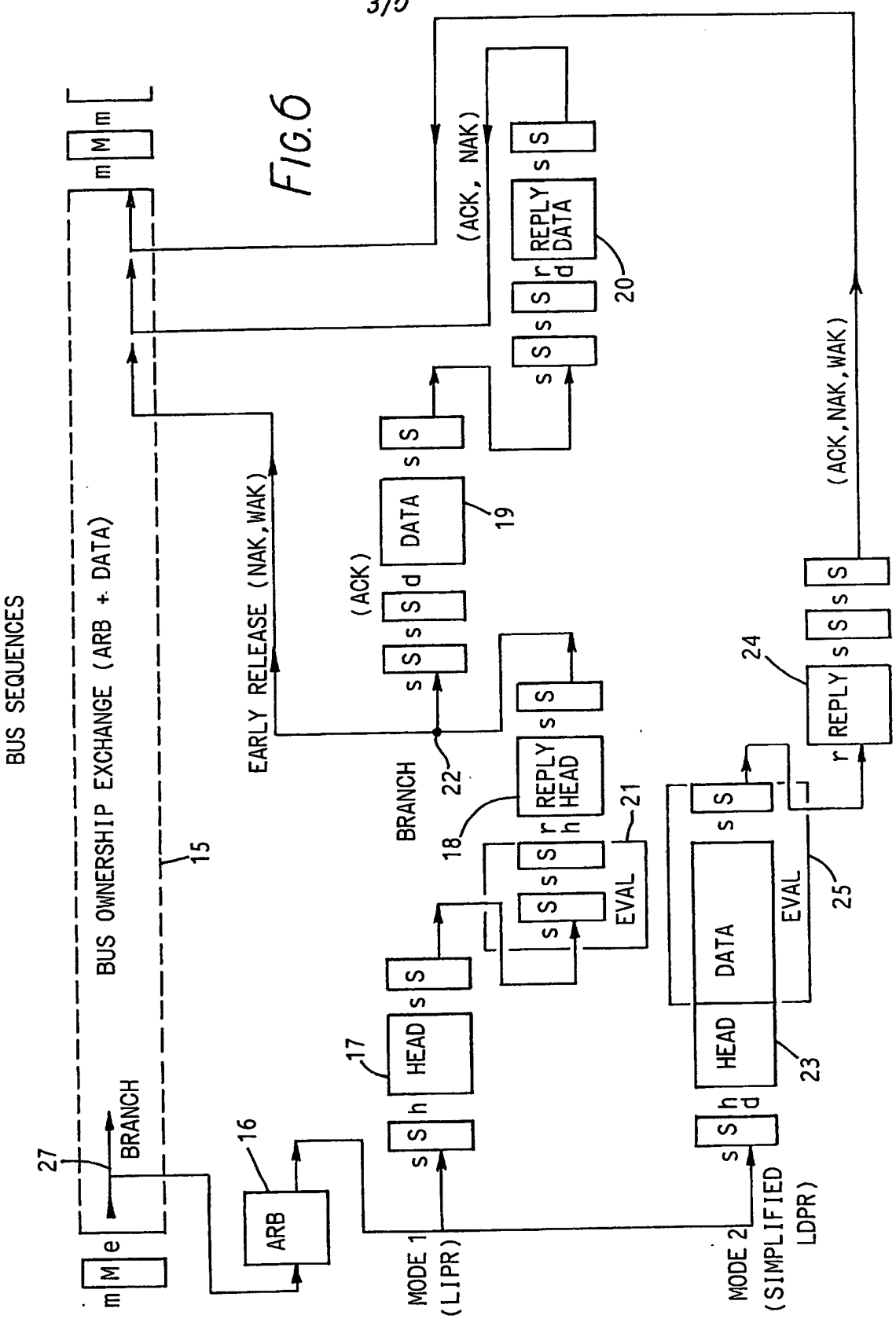

BUS SEQUENCES
BUS OWNERSHIP EXCHANGE (ARB + DATA)
BRANCH
ARB
MODE 1 (LIPR)
MODE 2 (SIMPLIFIED LDPR)
HEAD
DATA
EVAL
REPLY HEAD
REPLY DATA
EARLY RELEASE (NAK,WAK)
(ACK)
(ACK, NAK)
(ACK,NAK,WAK)
BRANCH
15
16
17
18
19
20
21
22
23
24
25
27

FIG.7

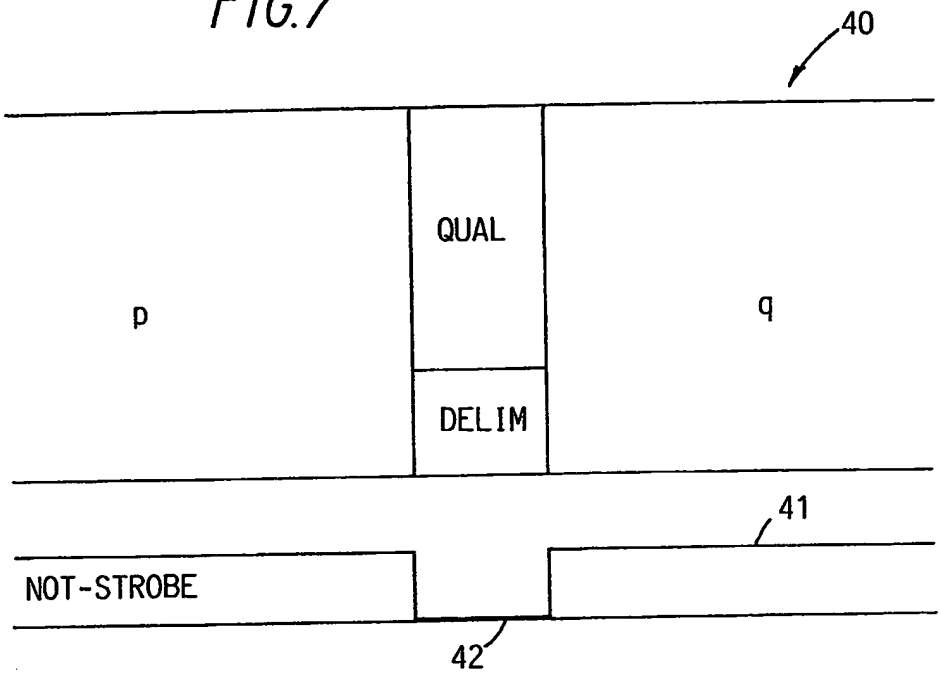
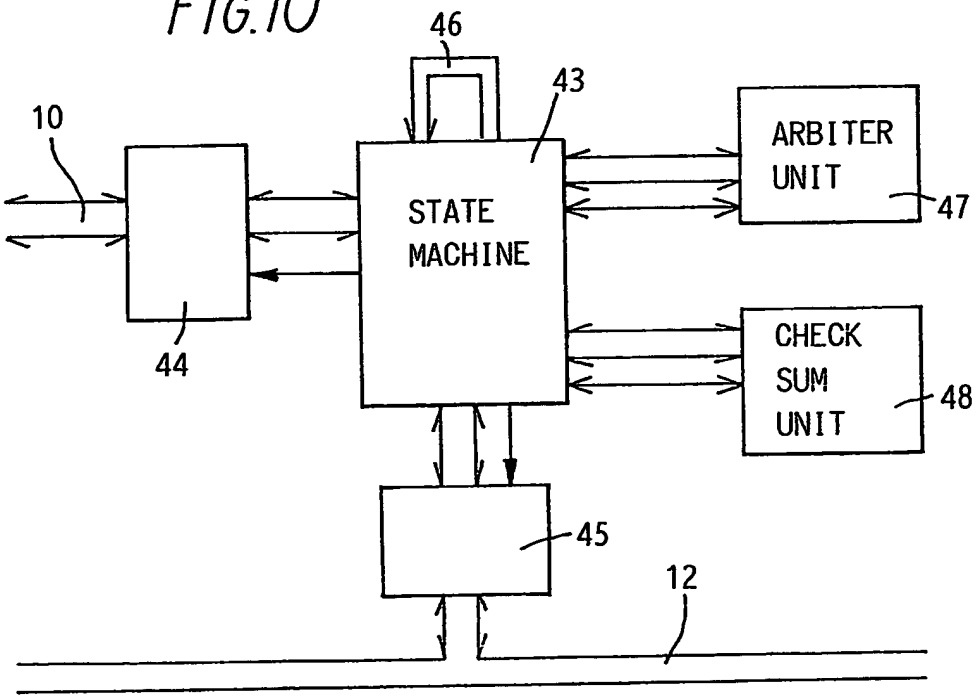

40
QUAL
p
q
DELIM
41
NOT-STROBE
42

FIG.10


46
43
10
ARBITER
UNIT
47
STATE
MACHINE
CHECK
SUM
UNIT
48
44
45
12

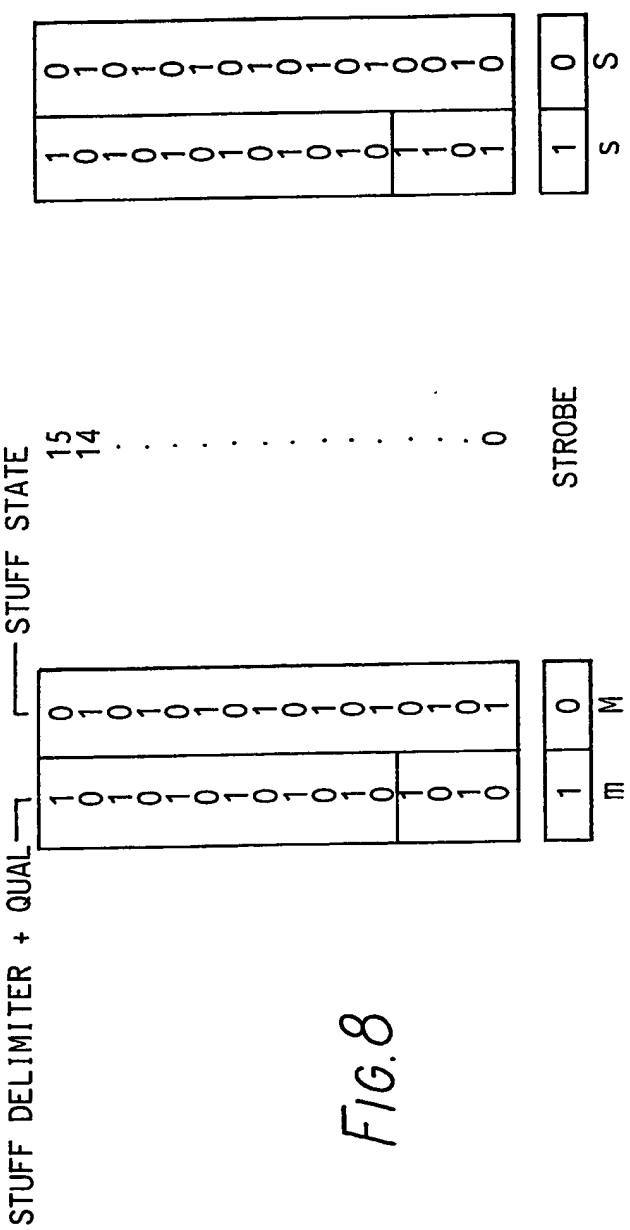

STUFF DELIMITER + QUAL
STUFF STATE
15
14
0
STROBE
m
M
s
S

FIG. 8

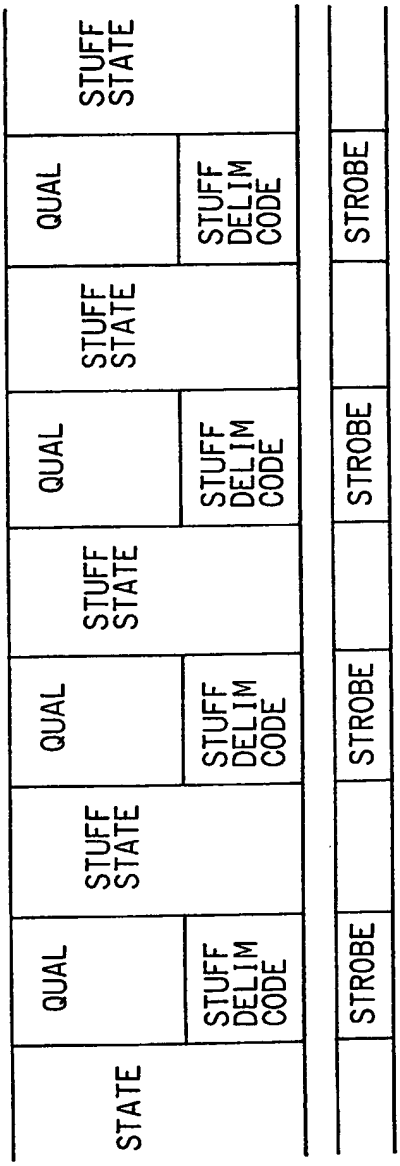

STATE
QUAL
STUFF DELIM CODE
STUFF STATE
STROBE

FIG. 9

COMPUTER COMMUNICATIONS BUS AND CONTROLLER AND PACKETS FOR COMMUNICATIONS THEREON

The present invention relates to a computer communications bus and controller, particularly but not exclusively a packet bus suitable for interconnecting computer systems, e.g. systems on different boards in a rack, and to a packet transfer system particularly suited for use therewith.

There exist many standard and proprietary computer buses, including various CPU buses and buses for connecting to peripheral devices, of which two examples are the IBM AT bus and the IEEE 488 bus. All such buses are characterized by large numbers of lines which may be divided into two main categories: information lines and control lines. Information lines may comprise data lines and address lines. Control lines are used to carry various mode, strobe and handshaking signals. In addition to these the bus may comprise power supply lines and/or one or more clock lines.

Proliferation of lines is undesirable from various points of view. The larger the number of lines the greater the risk of a transmission error. More lines are more expensive and large numbers of lines make it physically difficult to effect all connections in compactly designed equipment.

It is already known in computer memory buses to ameliorate this problem by using a single set of information lines in a multiplexed manner to carry both addresses and data. However such buses are slower (because the lines are shared) and still have large numbers of control lines.

Another problem when there are many control lines is that there are very many bus state changes which can occur. It is difficult to ensure that, in designing controllers, the bus always behaves in a determinate manner and that error conditions or exceptions are handled correctly.

The object of a first aspect of the present invention is to simplify the bus greatly and overcome the problems explained above.

The bus and controller according to the first aspect of the invention are defined in claims 1 and 11 below and

advantageous developments of the invention are defined in the claims dependent therefrom. It may nevertheless be noted briefly here that the information lines are used to carry not only delimiter codes and data but can also carry arbitration signals. It is thus possible to handle bus arbitration in a very secure yet simple way.

Packet communications on the bus and controller of the first aspect of the invention may use conventional packet techniques.

A conventional packet comprises a packet header frame PH followed by a packet data frame PD, each concluding with a check-sum CK which may be generated by the packet handler itself. Conventionally, a system or controller which is currently the bus master sends out a packet, each system on the bus reads the header to identify whether it is to receive the packet, and if so reads the packet data. The transmitting system, that is the bus master, cannot tell immediately whether the packet data has been accepted by the correct receiver(s) or the data read correctly. Error handling can only be achieved at a higher processing level when the receiver host system discovers an error in the received data. It may then transmit a packet back to the data transmitter requesting re-transmission.

The object of a second aspect of the invention is to overcome these disadvantages, reduce wastage of bus time and improve the reliability and error tolerance of packet communications.

The second aspect of the invention provides a method of communication by packet transfer as defined in claim 29 below. Advantageous features are defined in the claims dependent therefrom. A preferred system described in more detail below provides communication by packet transfer on a bus between a transmitter and a receiver each comprising a host system or processor and a bus controller, each packet comprising a header frame, optionally a data frame and at least one reply frame, with a reply frame being generated by the bus controller of the receiver without higher level host intervention and sent on the bus to the transmitter system, and the reply frame indicating to the transmitter system, the integrity of the preceding header

and/or data frame transmitted thereby, and/or the receive status of the receiver.

When using such connection-oriented packets a transmitter system can immediately be certain of the outcome of the packet transfer with no requirement for higher level host system intervention. The replies may be generated in most cases by the bus controller or packet handler of a receiving system before passing received data to its host system.

Reply frames may transmit various types of information from the packet receiver to the packet transmitter, and may be sent after the header frame and/or after the data frame. A packet may in some cases only comprise a header frame. The reply frame comprises at least an indication of the integrity of the preceding header and/or data, based on checksums transmitted by the transmitter with the packet, but in addition a reply frame sent after a header is received and before a data frame is transmitted may indicate that the receiver is ready to receive the data frame. The header may for example detail the length of the data frame to follow, enabling the receiver to evaluate whether it has a buffer and sufficient memory free to receive the data. A reply to a header may also indicate whether the connection queues at the receiver are full or whether the transmitter-receiver connection is valid and may under certain circumstances indicate that the transmitter should simply delay sending the data rather than immediately sending or aborting the packet.

Reply frames sent after the data frame of a packet may indicate for example the integrity of the data received, based on a checksum, and/or whether the receiver system has received the data successfully.

Since the controller and bus according to the first aspect of the invention are particularly well suited to data transfers using the communication oriented packets of the second aspect of the invention, the invention further relates to a bus and controller for using connection oriented packets as described.

All the various features of the invention will be explained in detail, by way of example, with reference to the

accompanying drawings in which:-

Figure 1 shows the structure of a conventional packet;

Figure 2 shows the structure of a short, auto reply (SAR) packet;

Figure 3 shows the structure of a long, deferred processed reply (LDPR) packet;

Figure 4 shows the structure of a long, immediate processed reply (LIPR) packet;

Figure 5 shows the packet bus and packet handler;

Figure 6 shows two alternative sequences of state changes involved in a system becoming bus master and sending a connection-oriented packet;

Figure 7 illustrates the handling of state changes;

Figure 8 illustrates the two "stuff" states;

Figure 9 illustrates the idle state of the bus; and

Figure 10 shows the structure of the packet handler.

A conventional packet is shown for reference in Figure 1, and comprises two frames, namely a packet header frame, PH, and a packet data frame, PD, each concluding with a checksum, CK.

Connection-Oriented Packets

We propose using instead of the conventional packet several possible types of what we shall refer to as connection-oriented packets, which will now be described.

(1) SAR Packet (Short, Auto Reply)

The Short, Auto Reply packet, Figure 2, consists of a variable length header (up to 16 words including mandatory connection information) and an automatic reply which is generated by the receiver. The reply is based on the header checksum only and is generated by the receiver packet handler or bus controller. This type of packet is intended for passing management data between cards on a packet bus and as such the general data capacity is low. The packet contains no separate packet data frame.

(2) LDPR Packet (Long, Deferred Processed Reply)

The Long, Deferred Processed Reply (LDPR) packet, Figure 3, consists of a variable length header followed by an automatic reply, a data frame and a processed reply. This is intended to be the primary packet type of the preferred packet bus embodying the invention and is at its most efficient for long data frame lengths. The first reply (after the Header frame) is generated automatically by the receiver packet handler and the packet can be aborted or delayed at this point if the reply is bad or erroneous. The reply will be bad either if the checksum of the header was wrong or if the receiver is unable to receive the data for any reason. For example, the header may detail the length of the data in the packet so that the receiver can assess whether it has sufficient memory available at that time to read the packet data.

The second reply (after the data frame) is a processed reply based on a status word supplied by the receiving system. The processing begins when the header is received and may continue during the receipt of the data frame. The processed reply (deferred) can then be sent after the data frame with minimal wastage of bus time.

In a simpler version of this packet type, the data frame may be sent immediately after the header, with no auto reply being sent. The deferred processed reply is still sent however to ensure that the packet has been correctly received. This packet type is discussed below with reference to Figure 6.

(3) LIPR Packet (Long, Immediate Processed Reply)

The Long, Immediate Processed Reply (LIPR) packet, Figure 4, comprises four frames similar to those in the LDPR packet. It differs however in that the first reply (after the header frame) is processed and the second (after the data frame) is automatic. The packet may therefore be terminated or delayed before the data frame is transmitted if the connection queues at the receiver are full or if the connection is invalid, in addition to the other reasons outlined above for the LDPR packet. However, the transmitter cannot send the data frame until a

processed reply to the header is sent from the receiver, which means that further communications on the bus can be delayed by any lengthy wait for a receiver system response.

The second reply, after transmission of the data frame, is automatic and refers only to the integrity of the data frame.

Connectionless Services

If the destination address in a packet is set to a predetermined broadcast address, e.g. 00, then all receiver systems read the packet. Since only one system may transmit on the bus at a time, all the receivers of a broadcast packet cannot generate and send reply packets, and so the broadcast packet must be "connectionless".

A host system may transmit a test packet in order to test its bus drivers and the bus. This packet is also connectionless.

Conventional connectionless packet transfers may also be used by the controller and bus embodying the invention. These may be either Short, No Reply (SNR) packets (header only), or Long, (LNR) packets (header and data). A conventional LNR packet is shown in Figure 1. Error handling with these packets must be handled at higher levels in transmitter and receiver systems but connectionless packet transfers are still of use for e.g. broadcast services and test services.

Packet Details

Frame integrity is ensured by means of a frame checksum transmitted in the last two words of each frame. Rather than a simple two-word overflowed checksum, an alternative technique which offers greater protection may be employed. This technique employs a running checksum of the checksum, in addition to the normal running checksum. Any overflows are discarded. The checksum words are appended to the frame as for conventional checksums.

The header frames, data frames and reply frames may be made up as follows.

(i) Header Frame

A packet always commences with a header frame sourced at the transmitter, and details at least the packet type and intended destination as well as a source identifier and a frame checksum. The header frame may for example contain the following fields:-

T-Field <0..2>: describes the present packet type.

= 000 Long Packet, Immediate Processed Reply (LIPR)

= 001 Long Packet, Deferred Processed Reply (LDPR)

= 010 Long Packet, No Reply (LNR)

= 011 Short Packet, Auto Reply (SAR)

= 100 Short Packet, No Reply (SNR)

G-Field <0...4> : Geographical (card) destination address. This field is supplied by the host processor but is mandatory.

C-Field <5...15> : Connection number at receiver.

Processor Sourced Words : These may comprise a packet data length indication and a data source identification (geographical and connection) in typical applications. Spare words may be used for user data in Short Packets.

CheckSum (sS) Words : Two word header checksum.

(ii) Data Frames

A data frame comprises from 1 to 512 words of user data followed immediately by two checksum words.

(iii) Reply Frames

Reply frames are generated by the receiver for the connection-oriented packet types. Each contains a single word of information and may be followed by a one or two word checksum. Two words of checksum may be more than is required but may be maintained nevertheless for consistency with other packet types to simplify system design.

The reply frame may for example comprise the following fields:

A (Auto Reply) :
Bit 0 = 0 checksum good
= 1 checksum bad

Bit 1 = 0 receiver ready
= 1 receiver not ready

P-Field (Processor Sourced Reply Information): This field is used in connection-oriented frames in which host processor intervention is required, e.g. to check that buffer space is available for an incoming packet and then to signal the result to the transmitter.

The Communications Bus and Controller

In the embodiment of the invention described below, the controller is a packet handler which controls the placement of packets on a bus and the reception of packets. Each packet is composed of word-serial, 16-bit-parallel data and the packet bus has only 16 information lines. A complete packet may consist of 64 words. As described above, bus performance is improved by the use of connection-oriented packets in accordance with the invention although conventional packets may also be used.

Figure 5 shows a 16-bit data bus 10 in communication with the packet handler 11 and acting as the source or destination of packet words. The packet handler 11 also communicates with a packet bus 12 which consists of 18 lines, namely 16 information lines INF0 to INF15, a single control line denoted NOT-STROBE and a clock line CLOCK.

The bus is in either a main idle state or a bus ownership exchange state (Figure 6). The main idle state consists of main "stuffing" states M lasting one clockcycle alternating with state changes m to state M. This is explained more fully below but its purpose is to keep the bus and current bus master constantly exercised. The main idle state terminates with a state change "e" denoting "exchange bus". Arbitration 16 then takes place

following which the new bus master controls the sending of a packet. Two alternative modes of message handling, using LDPR packets, are illustrated in Figure 6 and denoted MODE 1 and MODE 2. In MODE 1 a sub-idle state alternates with various active sub-states HEADer 17, REPLY HEADer 18, DATA 19 and REPLY DATA 20, to which entry is made by state changes h, rh, d and rd respectively. The sub-idle state consists of 1-clock sub-stuff states S alternating with state changes s to state S. There are thus seven state changes in all (m, e, s, h, rh, d, rd) and these could be signalled using just three bits.

Figure 5 shows the way in which the sixteen information lines are used in multiplexed manner to carry (1) data denoted DATA0-15, (2) four delimiter bits DELIM0-3 plus twelve qualifier bits QUALIF0-11, (3) arbitration bits denoted ARB0-ARB5, ARB+, ARB++ and PRI0-7.

The arbitration bits will be explained below. The data bits carry the packet headers, replies and data. The four delimiter bits DELIM0-3 carry delimiter codes for the state changes m, e, etc. Four bits are provided to allow for more than the seven state changes listed above. The twelve qualifier bits QUALIF0-11 carry a unique common code which must be present when NOT-STROBE goes low to assert a change of state for a change to be implemented, i.e. for a strobe to be validated. This guards against spurious changes of state when there is a noise pulse on the NOT-STROBE line, e.g. when there is a live board exchange in the rack.

The way in which state changes are effected is illustrated in Figure 7. The bus information lines are indicated at 40 and the waveform on the NOT-STROBE line at 41. The bus is initially in state p, representing any one of the states identified above. When it is required to change to the next state q, NOT-STROBE is pulsed low, as indicated at 42 and simultaneously the DELIMiter code for state q is placed on the lower four bits of the bus 40. The unique QUALifier code is placed on the upper twelve bits of the bus 40 to validate the strobe. The bus state changes to q. The DELIMiter code will always be m or s at the end of an active state, so switching to one of the idle states.

The "stuff" states M and S terminate all active bus states and provide a known idle condition. A stuff state is only 1 clock cycle in duration and must be followed by another stuff state delimiter or another state delimiter calling for an active state. This provides a means of determining that the bus is healthy and it is arranged that, as far as possible, all bit lines are exercised. The main stuff state M has the code pattern 1010101010101010 on the sixteen lines INF0 to INF15. Strobe is simultaneously not asserted - indicated as 0 in Figure 8, corresponding to NOT-STROBE high. The code m for switching to M is the exact complement of M. The bottom four bits 0101 are the DELIMiter code for m while the top twelve bits 010101010101 are the unique qualifier used with all delimiters. The stuff-state state S differs from the main stuff state M in that the bottom four bits are 0010 instead of 0101. The code s for switching to S is 1101, i.e. s is the exact complement of S.

Figure 9 shows the bus idling, in either stuff state with stuff states (M or S) alternating with stuff-state delimiters (m or s). The durations of all idling bus states are fixed and known to all controllers which can therefore predict when they will end.

In Figure 6, the bus is initially in a main stuff-state idling condition, the current bus master placing alternate m delimiters and M states on the bus. The or each prospective bus master wishing to gain ownership of the bus for the purpose of transmitting a data packet recognises this condition and, anticipating a further main stuff state delimiter m from the current bus master, drives the bus owner exchange delimiter e onto the bus at the same time. The delimiter e consists of an all 0's code which, by virtue of the wired-OR combination, overrides the main stuff state delimiter. The presence of the delimiter e enables the arbitration state 16.

In principle any controller may act as the source for the delimiter codes which define the state changes. The current bus master may act as the source of delimiters in idle periods or at least some of the controllers may place delimiters on the bus in sequence, e.g. during the main idle state. This is possible firstly because of the wired-OR nature of the bus and secondly

because at least the M and S states are of known duration (one clock period in the example described). All controllers can predict when such states will end and place a valid delimiter on the bus at the right time, e.g. an overriding "e" delimiter. Such a placement of an alternative valid delimiter forces a branch in the sequence of states, as at 22 and 27 in Figure 6.

Reverting to Figure 6, the bus states for a complete packet transfer can now be considered in detail. The bus is initially in the main idle state, until the current bus master places the delimiter e on the bus to exchange the bus, which starts by enabling the arbitration state 16. A system which wishes to participate in arbitration places its arbitration code on the bus. At the simplest level this may consist only of the bits ARB0 to ARB5 and arbitration can take place on the basis of any known arbitration algorithm.

The arbitration method may be based on the Futurebus scheme which allows a number of cards with mutually exclusive priorities to arbitrate for access to the bus. The circuit requires that at a predetermined point, all cards wishing to compete will place their priority number on the arbitration bus. The arbitration bus is a six line open-collector, negative logic bus. If any one card drives a line on this bus low (i.e. asserts it) the line will stay low. Thus if all competing cards drive their priority onto the bus and then observe the bus it is clear that all cards other than the highest priority card will see a bus value greater than theirs. These cards are then required to remove their priorities from the bus, ultimately leaving only that of the winning card.

Preferably ARB0 to ARB5 comprise the slot number or address for the system card and use is made of the remaining ten information lines INF7 to INF15 in an extended arbitration scheme as follows. At least some of the lines PRI0 to PRI7 (Figure 5) are used to carry a priority indicator independent of the slot address. It is then possible to implement priority arbitration. Moreover, in order to implement known algorithms which prevent a high priority system shutting out lower priority systems, the bits ARB+ and ARB++ may be used by a system to indicate that it has lost one or more arbitrations, thereby increasing its

priority and ensuring that it cannot be shut out. Fair access arbitration schemes of this nature are known per se, e.g. in the proprietary Multibus and Futurebus schemes, with the advantages of distributed arbitration (not reliant on a central arbiter) and support for extended priority. In distinction from the known schemes, the present invention allows the arbitration lines to be represented by the multiplexed usage of the information lines INFO-15 used for the data path. There is no requirement for separate "busy" and "compete" lines.

This is possible because of the carefully sequenced operation of the packet handler which deals also with the arbitration in a tightly controlled sequence of state changes from which any deviation represents an exception requiring higher level intervention.

Reverting yet again to Figure 6, the new bus master keeps the bus in its idle state s, S, s, S until it is ready to send a packet, which in this example (MODE 1 in Figure 6) is an LIPR packet. First, the bus master asserts the delimiter h and sends out the packet HEADer 17. All other systems on the bus respond to h to evaluate the header, to see if it is addressed to them, block 21, EVALuate. The system which is addressed in the header puts the delimiter rh on the bus and sends back the reply header which may be ACKnowledge, Not AcKnowledge, or WaitAcKnowledge. The bus master system can execute a branch, at 22. In the case of ACK it goes on to send the packet data 19. The addressed system puts rd on the bus and then sends the reply data 20, which will indicate for an LIPR packet only the integrity of the received data based on the data checksum. The bus master then restores the bus to the idle state.

It will be noted that there is only one possible branch point 22 (which leads to early release in the case of NAK or WAK) so that there is as far as possible a fixed sequence of states and virtually no scope for getting into confused states.

An alternative, simpler mode of operation (MODE 2 of Figure 6) involves the simplified LDPR packet described above. In this packet type the header and data are no longer separated and subject to separate replies but are joined and subject to a single reply applicable to both parts. Accordingly, the state

delimiter code hd defines a transition to a combined header/data state 23 and the state delimiter code r represents the reply state 24. The header is nevertheless distinct from the data and is delimited from the data part by its own checksum. This allows the EVALuation process 25 to proceed as soon as the header has been received, concurrently with the reception of the data part. It can be seen that for short data packets the evaluation time may be of near equivalence to the data part transmission time, making this mode of operation the optimum choice for data packets comprising short data frames.

For longer data packets a full LDPR packet may be used. The bus master then asserts the delimiter h and sends out the packet HEADer as in the LIPR packet (MODE 1) described above. The header contains a code defining the packet as a LDPR packet, so when the receiver system reads the header, the receiver packet handler evaluates the header checksum and sends an automatic reply indicating the integrity of the header and its own readiness to receive the packet data. As for the LIPR packet (MODE 1) a branch can then be executed by the bus master depending on the reply (ACK, NAK, WAK). For ACK, the bus master puts the delimiter d on the bus and sends the packet data. During receipt of the data, the receiver system can evaluate a processed reply. Once the data frame checksum has been received, the integrity of the data is checked by the receiver, and a processed reply sent to the bus master to confirm correct receipt of the packet. Time wastage on the bus is minimised in this way as the processed reply may be evaluated, at least in part, during receipt of the data frame, and sent with minimum delay thereafter.

Figure 10 is a block diagram of the packet handler 11 of Figure 5. The heart of the handler is a state machine 43 which receives signals on a data bus 10 and on the packet bus 12, via buffers 44 and 45, and also the signals representing its own current state as symbolised at 46. The state machine 43 is constructed in the manner well known in computer systems as combinatorial logic which, for all possible input state combinations, determines the output state. The output state is represented by signals put on the data bus 10 and on the packet

bus 12 via the buffers 44, 45, signals controlling the buffers 44 and 45, and signals controlling operation of an arbiter unit 47 and a check-sum unit 48 which are constructed in manner known per se.

Although not separately shown, the packet handler includes conventional means for comparing destination addresses with its own addresses, means for sending back ACK, NAK, WAK as appropriate and means for sending back an appropriate reply data 20 at the end of a packet.

In a controller according to the invention a heirarchy of states may be represented as described above, and at any particular level within the heirarchy a state may comprise one or more functionally associated sub-states together with the delimiter codes defining the transitions to those sub-states. In a preferred embodiment a controller then comprises a passive monitor for monitoring the sub-states applicable to any such heirarchical level regardless of other states or sub-states which may be present on the bus between those sub-states.

Such a passive monitor further enhances the error tolerance of the communications system and controller of the invention, but is particularly advantageous if, in cases where a sub-state sequence is predictable, it determines that a correct sequence of sub-states takes place at the heirarchical level monitored. The passive monitor can also monitor the elapsed time between such sub-state transitions for comparison with defined timeout periods.

CLAIMS:

1. A computer communications controller for coupling to a communications bus for controlling the writing of signals to and the reading of signals from the bus, the bus comprising a plurality of parallel information lines and a control line, and the controller being constructed as a state machine, each state representing a particular mode of communications on the bus, the controller signalling a change of state by placing a strobe signal on the control line, and concurrently placing on at least a sub-set of the information lines a delimiter code defining the change of state.

2. A controller according to claim 1, in which the delimiter code is written to a first sub-set of the information lines, and a qualifier code is written to a second sub-set of the information lines for confirming the integrity of the strobe signal.

3. A controller according to claim 2, in which the qualifier code is the same for every delimiter code.

4. A controller according to claim 1, 2 or 3, in which the states include active states and at least one idle, or stuff, state.

5. A controller according to claim 4, in which the or each idle state is a state of one clock cycle duration alternating with a delimiter code signalling the change to that idle state.

6. A controller according to claim 4 or 5, in which the binary condition of the information lines and control line defining the change to an idle state is the binary complement of the same lines in the idle state.

7. A controller according to any of claims 1 to 6, in which one of the states is an arbitration state in which the information lines carry arbitration signals.

8. A controller according to any of claims 1 to 7, in which a hierarchy of states is represented in which, at any particular level within the hierarchy, a state comprises one or more functionally associated sub-states together with the delimiter codes defining the transitions to those sub-states, the controller comprising a passive monitor for monitoring the sub-states applicable to at least one hierarchical level regardless of other states or sub-states which may be present on the bus between those sub-states.

9. A controller according to claim 8, in which the passive monitor determines that a correct sequence of sub-states takes place at the hierarchical level monitored, in cases where the sub-state sequence is predictable.

10. A controller according to claim 8 or 9, in which the passive monitor monitors the elapsed time between sub-state transitions, where the sub-state sequence is predictable, for comparison with defined timeout periods.

11. A computer communications system comprising a plurality of communications controllers coupled to a communications bus for controlling the writing and reading of signals to and from the bus, the bus comprising a plurality of parallel information lines and a control line, and each controller being constructed as a state machine, each state representing a particular mode of communication(s) on the bus, a controller signalling a change of state by placing a strobe signal on the control line, and concurrently placing on at least a sub-set of the information lines a delimiter code defining the change of state.

12. A system according to claim 11, in which the delimiter code is written to a first sub-set of the information lines, and a qualifier code is written to a second sub-set of the

information lines to enable checking of the integrity of the strobe signal by the controllers.

13. A system according to claim 12, in which the qualifier code is the same for every delimiter code.

14. A system according to any of claims 11, 12 or 13 in which the states include active states and at least one idle, or stuff, state.

15. A system according to claim 14, in which an idle period consists of repeated idle states, each idle state being a state of one clock cycle duration, alternating with a delimiter code signalling the change to that idle state.

16. A system according to claim 15, in which, during an idle period, successive repeats of an idle state and the delimiter code signalling the change to that idle state are written to the bus by successive controllers in a predetermined sequence of controllers, which sequence is known to the controllers.

17. A system according to any of claims 14 to 16 in which the signal on the bus signalling a change of state to an idle state is the binary complement of the signal on the bus during that idle state.

18. A system according to any of claims 11 to 17, in which at least some states exist for a fixed, predetermined number of clock periods, so that a controller can predict when any such state will end.

19. A system according to claim 18, in which a controller can take over control of the bus by predicting the end of a state of fixed duration and by writing an appropriate delimiter code onto the bus at the predicted end of the state of fixed duration.

20. A system according to claim 11 to 19, in which one of the states is an arbitration state in which the information lines carry arbitration signals.

21. A system according to any of claims 11 to 20, in which a delimiter code signalling a state change comprises a wired-OR combination of signals placed on the bus by a plurality of controllers.

22. A system according to claim 21, in which a delimiter code comprises all bits placed on the bus forced low by any one or more controllers, so that any other controller placing a delimiter code on the bus at that time can force a change to a desired state.

23. A system according to any of claims 11 to 22, in which a state comprises the transfer of word-serial information on the bus from a sending, or transmitting, controller to one or more receiving controllers.

24. A system according to any of claims 11 to 23, in which a hierarchy of states is represented, a state at an heirarchical level comprising one or more functionally associated sub-states and the delimiter codes for signalling transitions to those sub-states.

25. A system according to claim 24, in which a controller comprises a passive monitor for monitoring the states applicable to that heirarchical level regardless of other states which may exist on the bus between those states.

26. A system according to claim 24 or 25, in which the passive monitor monitors at least one specific heirarchical level at which the state sequence is predictable to determine that a correct sequence of states occurs.

27. A system according to claim 26, in which the passive monitor monitors the elapsed time between the state transitions at a heirarchical level for comparison with defined timeout periods.

28. A system according to any of claims 24 to 27, in which a state comprises a sequence of sub-states which constitutes an exchange of information between two controllers.

29. A system according to claim 28 in which one controller, the bus master or transmitter, is primarily an information transmitter and the other controller, the receiver, responds with replies dependent at least in part on information transmitted by the transmitter.

30. A method of communication by packet transfer on a bus between a transmitter system and a receiver system both coupled to the bus, a packet comprising a header frame and a reply frame, the header being transmitted by the transmitter and comprising a receiver address or identifier, and the reply being generated by the receiver in response to the header and sent by the receiver to the transmitter, the reply indicating that the receiver has received the header frame.

31. A method according to claim 30 in which the header frame comprises a checksum, to be used by the receiver to check the integrity of the received header frame and to indicate in the reply frame the integrity or otherwise of the received header frame.

32. A method accordng to claim 30 or 31 in which the transmitter system and the receiver system each comprise a packet controller coupled to the bus and a host processor coupled to the controller, the reply frame being generated by the receiver controller substantially without receiver host intervention.

33. A method according to any of claims 30 to 32 in which a packet comprises a header frame and a subsequent data frame, a

reply frame is generated and sent by the receiver in response to the header frame, the transmitter waits for the reply frame after transmitting the header frame and on the basis of the reply assesses whether to transmit the data frame immediately, to wait, or to abort the packet.

34. A method according to claim 33 in which the header reply frame indicates the integrity of the header as received, based on a header checksum, and indicates whether or not the receiver is able to receive the following data frame.

35. A method according to claim 34 in which the receiver can indicate its readiness to receive a data frame by signalling acknowledge (ACK), not acknowledge (NAK) or wait acknowledge (WAK).

36. A method according to claim 33 or 34 in which the header frame indicates the length of the following data frame to enable the receiver to assess whether it has sufficient memory or buffer space free to read the data frame.

37. A method according to any of claims 33 to 36 in which the packet further comprises a second reply frame, a data reply frame, sent by the receiver after the data frame.

38. A method according to claim 37 in which the data reply frame indicates the integrity of the received data frame, based on a data frame checksum transmitted at the end of the data frame.

39. A method according to claim 30 in which a packet further comprises a data frame transmitted immediately after the header frame, the reply frame being sent after receipt of the data frame.

Patents Act 1977

E: miner's report to the Comptroller under Section 17 (The Search Report)

Application number

9205094.7

Relevant Technical fields

(i) UK Cl (Edition) H4P-PPNC

(ii) Int CL (Edition) G06-13/14;H04L-12/40

Search Examiner

S J DAVIES

Databases (see over)

(i) UK Patent Office

(ii)

Date of Search

8 JUNE 1992

Documents considered relevant following a search in respect of claims 1-29

Category (see over)	Identity of document and relevant passages	Relevant to claim(s)
A	US 3978451 (ITO ET AL) column 3 - column 5, 1.21	

with present

ent for

Category	Identity of document and relevant passages	Relevant to claim(s)

Categories of documents

X: Document indicating lack of novelty or of inventive step.

Y: Document indicating lack of inventive step if combined with one or more other documents of the same category.

A: Document indicating technological background and/or state of the art.

P: Document published on or after the declared priority date but before the filing date of the present application.

E: Patent document published on or after, but with priority date earlier than, the filing date of the present application.

&: Member of the same patent family, corresponding document.

Databases: The UK Patent Office database comprises classified collections of GB, EP, WO and US patent specifications as outlined periodically in the Official Journal (Patents). The on-line databases considered for search are also listed periodically in the Official Journal (Patents).

Published 1993 at The Patent Office, Concept House, Cardiff Road, Newport, Gwent NP9 1RH. Further copies may be obtained from Sales Branch, Unit 6, Nine Mile Point, Cwmfelinfach, Cross Keys, Newport, NP1 7HZ. Printed by Multiplex techniques ltd, St Mary Cray, Kent.